

Fichier package.json

```
{
  "name": "bootstrap ",
  "version": "1.0.0",
  "scripts": {
    "tsc": "tsc",
    "tsc:w": "tsc -w",
    "lite": "lite-server",
    "start": "concurrent \"npm run tsc:w\" \"npm run lite\" "
  },
  "license": "ISC",
  "dependencies": {
    "angular2": "2.0.0-beta.0",
    "systemjs": "0.19.6",
    "es6-promise": "^3.0.2",
    "es6-shim": "^0.35.3",
    "reflect-metadata": "0.1.2",
    "rxjs": "5.0.0-beta.0",
    "zone.js": "0.5.10"
  },
  "devDependencies": {
    "concurrently": "^1.0.0",
    "lite-server": "^1.3.1",
    "typescript": "^1.7.3"
  }
}
```

Explications package.json

Ce fichier permet de charger les dépendances nécessaires pour le projet. On trouve dans ce fichier :

Les scripts disponibles :

- * tsc : Compilation du code TypeScript
- * tsc:w : Compilation du code TypeScript avec un watcher sur les changements
- * lite : Démarrage d'un serveur http lite
- * start : Démarrage de la compilation avec watcher et du serveur de manière concurrentielle

Les dépendances :

- * angular2
- * systemjs : Permet le chargement dynamique de module
- * es6-promise : Promises
- * es6-shim : Polyfills permettant d'émuler certaines fonctionnalités sur des navigateurs non compatibles
- * reflect-metadata : Permet d'ajouter des décorateurs
- * rxjs : Reactive Programming
- * zone.js : Equivalent des threads locaux en Java

Les dépendances de développement :



By **Kelem**
cheatography.com/kelem/

Published 13th January, 2016.
 Last updated 13th May, 2016.
 Page 1 of 3.

Sponsored by **Readable.com**
 Measure your website readability!
<https://readable.com>

Explications package.json (cont)

- * concurrently : Pour démarrer des scripts de manière concurrente
- * lite-server
- * typescript

Fichier tsconfig.json

```
{
  "compilerOptions": {
    "module": "system",
    "target": "es5",
    "moduleResolution": "node",
    "sourceMap": true,
    "emitDecoratorMetadata": true,
    "experimentalDecorators": true,
    "removeComments": false,
    "noImplicitAny": false
  },
  "exclude": [
    "node_modules"
  ]
}
```

Permet de spécifier des options pour la compilation du TypeScript en JavaScript. Voir les ressources suivantes :

<https://github.com/Microsoft/TypeScript/wiki/tsconfig.json>

<https://github.com/Microsoft/TypeScript/wiki/Compiler-Options>

Création d'un composant

```
import {Component} from "angular2/core";
@Component({
  selector: 'my-app',
  template: '<h 1>My first Angular 2 App</h 1>'
})
export class AppComponent {}
```

Les étapes sont :

- * Imports
- * Décorateurs
- * Classes

Fichier boot.ts

```
import {bootstrap} from "angular2/platform/browser";
import {AppComponent} from "./app.component";
bootstrap(AppComponent);
```

Ce fichier permet de *booter* notre application. On boote notre composant principal. Attention à bien utiliser angular2/platform/browser.



By Kelem
cheatography.com/kelem/

Published 13th January, 2016.
Last updated 13th May, 2016.
Page 2 of 3.

Sponsored by [Readable.com](https://readable.com)
Measure your website readability!
<https://readable.com>

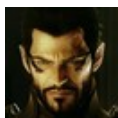
Fichier index.html

```
<html>
<head>
  <title>Angular 2 QuickStart</title>
  <!-- 1. Load libraries -->
  <script src="node_modules/angular-2/bundles/angular2-polyfills.js"></script>
  <script src="node_modules/systemjs/dist/system.src.js"></script>
  <script src="node_modules/rxjs/bundles/Rx.js"></script>
  <script src="node_modules/angular-2/bundles/angular2-declarations.js"></script>
  <!-- 2. Configure SystemJS -->
  <script>
    System.config({
      packages: {
        app: {
          format: 'register',
          defaultExtension: 'js'
        }
      }
    });
    System.import('app/boot')
      .then(null, console.error.bind(console));
  </script>
</head>
<!-- 3. Display the application -->
<body>
<my-app>Loading...</my-app>
</body>
</html>
```

- * Chargement des librairies nécessaires
- * Chargement du module via SystemJS
- * Affichage de notre composant

Récapitulatif

- On crée un nouveau projet vide dans Webstorm.
- On crée un fichier package.json pour les dépendances.
- On crée ensuite un tsconfig.json qui va permettre la compilation TypeScript en Javascript (Plus précisément la transpilation).
- On crée notre composant angular.
- On crée ensuite un fichier boot.ts pour gérer le bootstrap de notre application.
- On termine par la création de notre page index.html.



By **Kelem**
cheatography.com/kelem/

Published 13th January, 2016.
 Last updated 13th May, 2016.
 Page 3 of 3.

Sponsored by **Readable.com**
 Measure your website readability!
<https://readable.com>